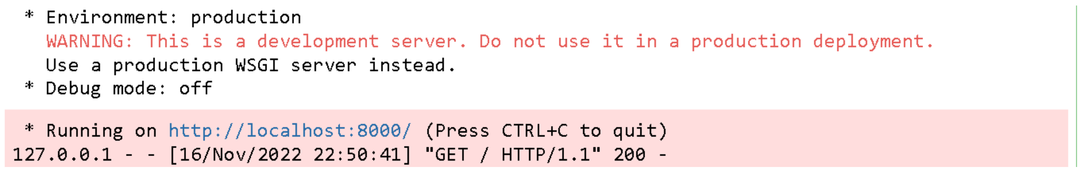
Project Development Phase
Sprint-3 Test Case

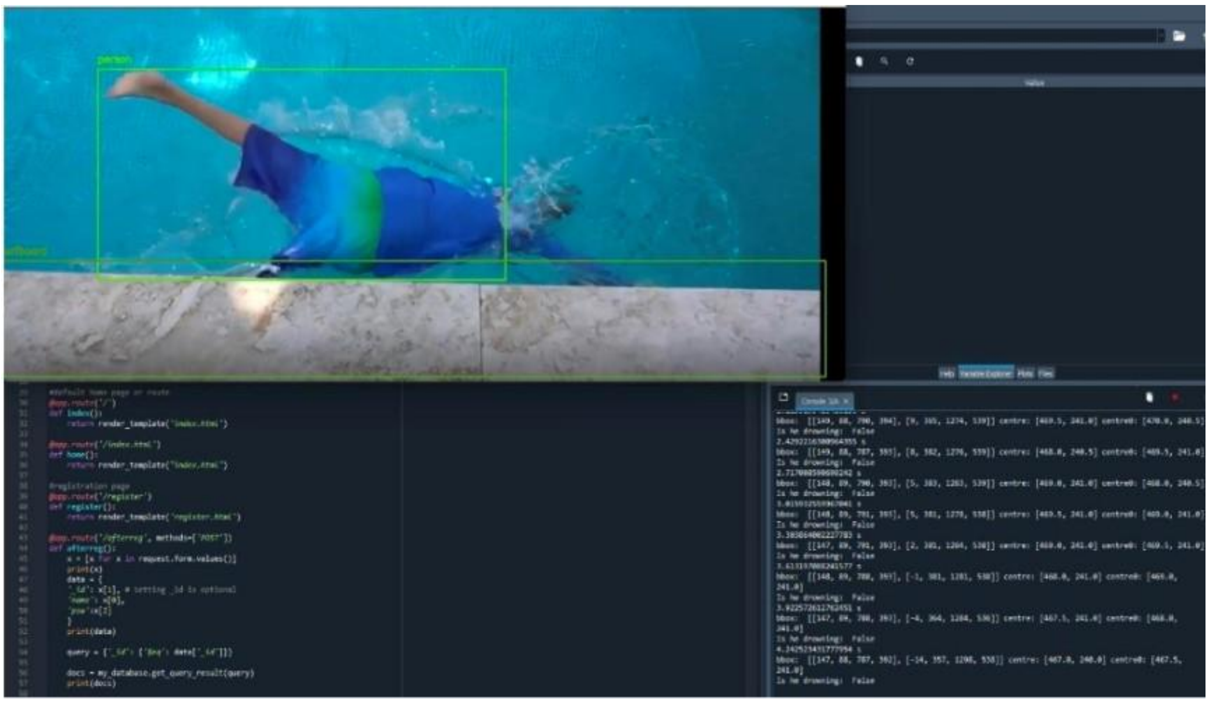
Date	**16 November 2022**
Team ID	**PNT2022TMID25362**
Project Name	**VirtualEye- Life Guard for Swimming Pools to Detect Active Drowning**
Maximum Marks	**8 Marks**

Source Code:

```
import cvlib as cv
import os
from cvlib.object_detection import draw_bbox
import cv2
import time
import numpy as np
from playsound import playsound
import requests
from flask import Flask, request, render_template, redirect, url_for
from cloudant.client import Cloudant
client=Cloudant.iam("eebf1e69-3eb8-4997-8323-01a9ad425481-
bluemix","sK07OEEKl3hRZ_hfXKUa-2LcvdzbTkEfezkG3Zn4Xyuw",connect=True)
my_database=client.create_database("my_database")

app=Flask(__name__)
@app.route('/')
def index():
    return render_template("index.html")

@app.route('/index.html')
def home():
    return render_template("index.html")


@app.route('/register.html')
def register():
    return render_template("register.html")
@app.route('/afterreg',methods=['POST'])
def afterreg():
    x = [x for x in request.form.values()]
    print(x)
    data={'_id': x[1],'name': x[0],'psw':x[2]}
    print(data)

    query = {'_id': {'Seq': data['id']}}

    docs = my_database.get_query_result(query)
    print(docs)

    print(len(docs.oll()))

    if(len(docs.all())--0):
        url = my_database.create_document (data)

        return render_template('register.html', pred="Registration Successful, please login using
your details")
    else:

        return render_template("register.html", pred="You are already a member, please login
using your detalls")

@app.route('/login')
def login():
    return render_template('login.html')
@app.route('/afterlogin',methods=['POST'])
def afterlogin():
    user = request.form['_id']
    passw = request.form['psw']
    print(user,passw)
    query ={'_id':{'Seq':user}}
    docs =my_database.get_query_result(query)
    print(docs)
    print(len(docs.all()))
    if(len(docs.all())==0):
        return render_template('login.html',pred='The username is not found.')
    else:
        if((user==docs[0][0]['_id']and passw==docs[0][0]['psw'])):
            return redirect(url_for('prediction'))
        else:
            print('Invalid User')
def logout ():
    return render_template('Logout.html')
@app.route('/result',methods=["GET","POST"])
def res():
    webcam=cv2.VideoCapture('drowning.mp4')
    if not webcam.isOpened():
        print("Could not  open webcam")
        exit()
    t0=time.time()
    centre0 = np.zeros(2)
    isDrowning =False
    frame=webcam.read()
    while webcam.isOpened():
        status,frame=webcam.read()
        bbox,label,conf= cv.detect_common_objects(frame)
        if(len(bbox)>0):
            bbox0=bbox[0]
            centre =[0,0]
            centre=[(bbox0[0]+bbox[2])/2,(bbox0[1]+bbox0[3])/2]
            hmov=abs(centre[0]-centre0[0])
            vmov=abs(centre[1]-centre0[1])
        x=time.time()
        thershold=10
        if(hmov>threshold or vmov>threshold):
            print(x-t0,'s')
            t0=time.time()
            isDrowning = False
        else:
            print(x-t0,'s')
            if((time.time()-t0)>10):
                isDrowning=True
        print('bb0x: ',bbox,'centre:',centre,'centre0:',centre0)
        print('Is he drowning:',isDrowning)
        centre0=centre
        out=draw_bbox(frame,bbox,label,conf,isDrowning)
        cv2.imshow("Real-time object detection".out)
        if(isDrowning == True):
            playsound('alarm.mp3')
            webcam.release()
            cv2.destroyAllWindows()
            return render_template('prediction.html',prediction="Emergency !!! The person is
drowning")
        if cxv2.waitkey(1) & 0xFF == ord('q'):
            break
        webcam.release()
        cv2.destroyAllWindows()
if __name__=="__main__":
    app.run(host="localhost",port=8000,debug=False)
```



```
 * Environment: production
   WARNING: This is a development server. Do not use it in a production deployment.
   Use a production WSGI server instead.
 * Debug mode: off
 * Running on http://localhost:8000/ (Press CTRL+C to quit)
127.0.0.1 - - [16/Nov/2022 22:50:41] "GET / HTTP/1.1" 200 -
```



```
#default home page or route
@app.route('/')
def index():
    return render_template('index.html')

@app.route('/index.html')
def home():
    return render_template('index.html')

#registration page
@app.route('/register')
def register():
    return render_template('register.html')

@app.route('/afterreg', methods=['POST'])
def afterreg():
    x = [x for x in request.form.values()]
    print(x)
    data = {
    '_id': x[1], # setting _id is optional
    'name': x[0],
    'psw':x[2]
    }
    print(data)

    query = {'_id': {'$eq': data['_id']}}

    docs = my_database.get_query_result(query)
    print(docs)
```

```
bbox:  [[149, 88, 790, 394], [9, 385, 1274, 539]] centre: [469.5, 241.0] centre0: [470.0, 240.5]
Is he drowning:  False
2.4292216300964355 s
bbox:  [[149, 88, 787, 393], [8, 382, 1276, 539]] centre: [468.0, 240.5] centre0: [469.5, 241.0]
Is he drowning:  False
2.717000589000242 s
bbox:  [[148, 89, 790, 393], [5, 383, 1283, 539]] centre: [469.0, 241.0] centre0: [468.0, 240.5]
Is he drowning:  False
3.0159325599670061 s
bbox:  [[148, 89, 791, 393], [5, 381, 1278, 538]] centre: [469.5, 241.0] centre0: [469.0, 241.0]
Is he drowning:  False
3.3038640082227783 s
bbox:  [[147, 89, 791, 393], [2, 381, 1284, 538]] centre: [469.0, 241.0] centre0: [469.5, 241.0]
Is he drowning:  False
3.6133970082415577 s
bbox:  [[148, 89, 788, 393], [-1, 381, 1281, 538]] centre: [468.0, 241.0] centre0: [469.0,
241.0]
Is he drowning:  False
3.9225726122762451 s
bbox:  [[147, 89, 788, 393], [-4, 364, 1284, 536]] centre: [467.5, 241.0] centre0: [468.0,
241.0]
Is he drowning:  False
4.2425234317777954 s
bbox:  [[147, 88, 787, 392], [-14, 357, 1298, 538]] centre: [467.0, 240.0] centre0: [467.5,
241.0]
Is he drowning:  False
```